ETHME – Status Update IV

ETHME time & task recording

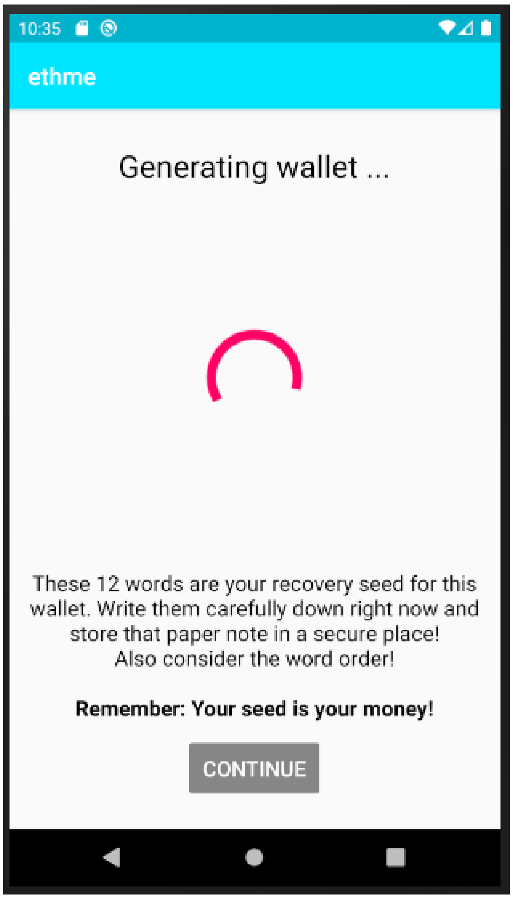
project workload	125,00 h	**team size**	1 persons
per person	125,00 h		
		start	23.9.2019
workload left over	35,50 h	**end**	28.10.2019
per person	35,50 h	**duration**	5 weeks
left h per week / person	**35,50 h**	**today**	21.10.2019
per working day	**7,10 h**	**till end**	**1 weeks**
Projects GitHub:	https://github.com/arminbrandy/ethme	**total**	**89,50 h**

Armin Brandy		
task - *commit*	**hours**	**date**
Research & evaluation	4,00 h	23.09.2019
Defining and writing the Vision as "Whitepaper" - *look at init commit - *ebbcc22*	2,00 h	23.09.2019
Start reading Mastering Ethereum	3,00 h	26.09.2019
Setup Project, basic environment & GitHub	3,00 h	27.09.2019
FH Blockchain LV - more about blockchain	3,00 h	27.09.2019
Starting Project, more environment & creating basic App - **4b636ea*	4,00 h	28.09.2019
ETHME - Status Update I	1,25 h	28.09.2019
Working on connecting to node via web3j and on Wallet creation...	4,50 h	29.09.2019
Successfully testing first web3j Wallet functionalities with web3j doc	4,50 h	02.10.2019
Learning about Wallet standards as BIP39/44 and related, Javas 'true' SecureRandom fns Exploring & testing wallet creation & storage with web3j and understanding the concepts	7,50 h	03.10.2019
Concept of user auth to access wallet & PIN activity for user authentication	2,75 h	04.10.2019
FH Blockchain LV - more into Ethereum & smart contracts	3,25 h	04.10.2019
Design PIN activity & designing secure storage structure	4,00 h	05.10.2019
ETHME - Status Update II	1,50 h	05.10.2019
Some redesign, PinPad activity & exact wallet encryption concept via PIN & Fingerprint	7,25 h	08.10.2019
Defined user & data flow of whole wallet creation process	4,25 h	10.10.2019
Finishing PinPad setup functionality - **4af0a1d*	5,25 h	11.10.2019
ETHME - Status Update III	1,50 h	11.10.2019
Checking out & evaluating https://github.com/android/security-samples	1,50 h	13.10.2019
Proof of concept for PIN -> Keystore.sign() -> pw for Wallet.creation() - **b28a1df*	5,00 h	15.10.2019
Wallet.class & displaying mnemonic seed, organizing creation functionality	8,50 h	17.10.2019
FH Blockchain LV - UE smart contract learning	2,00 h	18.10.2019
FH Blockchain LV - Ethereum, smart contracts, dev enviroment & ERC Standards	3,00 h	18.10.2019
Found compatibility problem with wallet standards BIP39/44 ... try debugging	1,50 h	19.10.2019
ETHME - Status Update IV	1,50 h	21.10.2019

Progress

Last week, finally a proof of concept was implemented for the wallet creation process, which is visible in the commit **b28a1df.* It shows that all the technologies work together as defined in the previous status update and created an encrypted wallet file, using an Android KeyStores key for encryption.
After proofing this functionality, the user flow of actually creating the wallet, with all in between steps and a clean codebase was implemented and is about to be finished.
This means, everything is implemented including displaying the Mnemonic seed (see screenshots below). Solely the processes last activity, which asks the user for 4 randomly selected words of its newly generated seed, in order to check, that he actually took note of its seed still needs to be implemented. This won't be that hard of a task.

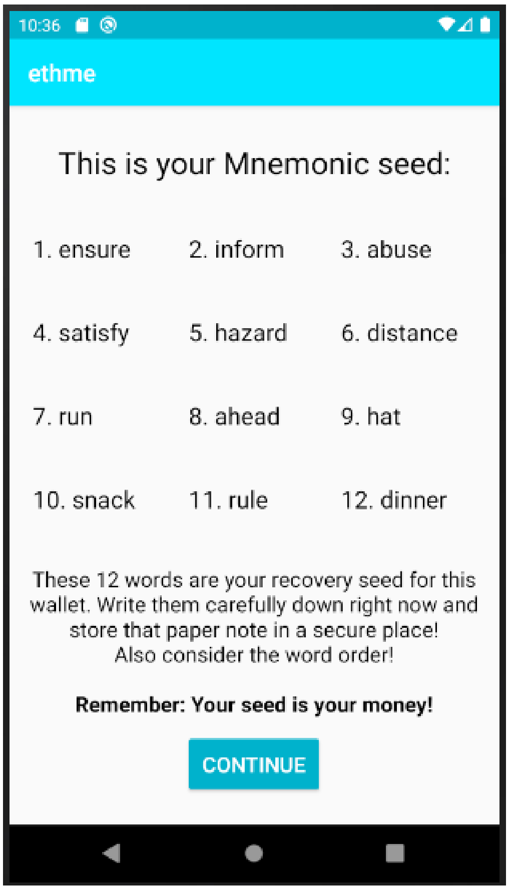
Talking in Code, we have implemented two pretty clean Wallet classes, one Activity and one pure Wallet functionality, which take care of all the Wallet actions needed for creating & also accessing the wallet.
In order to conclude the whole process, we still need to store the meta information, about the new wallet files name and it's corresponding Android Key somewhere. The Androids Shared preferences or a simple json File, seem to be enough for that. This last task comes with above's last seed checking activity.



Generating wallet ...

These 12 words are your recovery seed for this wallet. Write them carefully down right now and store that paper note in a secure place!
Also consider the word order!

Remember: Your seed is your money!



This is your Mnemonic seed:

1. ensure 2. inform 3. abuse
4. satisfy 5. hazard 6. distance
7. run 8. ahead 9. hat
10. snack 11. rule 12. dinner

These 12 words are your recovery seed for this wallet. Write them carefully down right now and store that paper note in a secure place!
Also consider the word order!

Remember: Your seed is your money!

A kind of critical Bug …

Besides this nice development, one crucial bug in above's wallet creation process was found last Friday. Regarding the seed generation, or better the private key derivation.
Somehow the web3j library's methods, in the current implementation of ETHME, don't behave like expected and therefore derivate a different private key and hence a different ethereum address from the generated mnemonic seed, as the standards suggest.
This was discovered by testing ethmes generated seeds with other wallets, as chromes browser extension MetaMask and the Android App Jwallet and comparing their derivated ethereum address with ours. They deviated.

What does this mean?

Until this bug isn't fixed, it'd mean that the seed only works with our current exact implementation and fails in fulfilling its requirement to be a standardized backup method of the corresponding ethereum wallet, since you can't recover your wallet anywhere else except on exactly currents version of ETHME. Short, the wallet would be useless like this.

However, regarding the development, it actually doesn't stop the App anyhow from working with that not standardized seed derivation. Since it's still a cryptographical valid object.

So how are we proceeding then?

This project has the demand of both following aspects:

- One being a standardized and securely working ethereum gateway.
- Two to satisfy all of its originally intended use cases.

Therefore, I'd suggest to give the debugging of the arised problem a try of a few working hours, if needed. Let's say one full working day, so 8h. And if by then, there's still no solution in hindsight. Continuing with the next tasks anyway, in order to get at least the desired functionality as far as possible.
With the task of resolving that critical issue after the basic functionalities of the mvp version are implemented.

Ps. This bug is also part of the reason for that delayed Status update, since it was found right before the update was supposed to be written. I tried to debug that first, but couldn't find a solution right away, so that somehow disrupted my time management and my idea of how to handle that bug right now. So, I took the time to postpone that update after the weekend.